# Fișa 09 C# – constructori și destructori

1. Un **constructor** este o metodă specială a unei clase care se apelează automat de câte ori se instanțiază (construiește) un obiect al clasei respective. Rolul constructorului este de a aloca memorie pentru câmpurile și metodele obiectului respectiv respectând declarațiile din clasă și, eventual, de a inițializa câmpurile respective. Dacă utilizatorul nu definește un constructor pentru o clasă acesta va fi creat automat de către sistem. Constructorul este o funcție care nu returnează nimic și are același nume ca și clasa din care face parte. Această funcție poate fi supraîncărcată și de obicei accesul la ea e public. Exemplu:

class Program

{

class Masina

{

public string marca, tip;

public int pret;

//constructor fără parametri care inițializează câmpurile cu valori implicite

public Masina()

{

marca = "necunoscută";

tip = "necunoscut";

pret = 0;

Console.Write("\nAm construit o mașină cu date implicite!");

}

public Masina(string m) //inițializare câmp marca

{

marca = m;

Console.Write("\nAm construit o mașină cu marca " + marca + "!");

}

public Masina(string m, string t) //inițializare câmpuri marca, tip

{

marca = m;

tip = t;

Console.Write("\nAm construit o mașină cu marca "+m+" și tipul "+t+"!");

}

public Masina(string m, string t, int p) //se inițializează toate câmpurile

{

Console.Write("\nAm construit o mașina "+m+" "+t+". Costa "+p+" dolari!");

}

}

static void Main(string[] args)

{

//setam setul de caractere la consola ca UTF8 - pentru afișare diacritice

Console.OutputEncoding = UTF8Encoding.UTF8;

//definim doua obiecte de tip Mașină

//nu se va afișa nimic: s-au definit doar referințe (nume), fără a se aloca memorie

Masina m1, m2, m3, m4;

//se vor apela pe rând cei 4 constructori, fiecare afișând un mesaj

m1 = new Masina();

m2 = new Masina("Mazda");

m3 = new Masina("Volkswagen", "Polo Happy");

m4 = new Masina("Lamborghini", "Veneno Roadster", 5414976);

//gata

Console.Write("\n\n\n");

}

}

Se va afișa:

**Am construit o mașină cu date implicite!**

**Am construit o mașină cu marca Mazda!**

**Am construit o mașină cu marca Volkswagen și tipul Polo Happy!**

**Am construit o mașina Lamborghini Veneno Roadster. Costa 5414976 dolari!**

**Press any key to continue . . .**

1. Un **destructor** este o metodă specială care se apelează când se distruge un obiect (se eliberează memoria ocupată de el). Metoda va avea același nume ca și clasa din care face parte, plus prefixul ~. O clasă poate avea un singur destructor. În cele mai multe cazuri definirea unui destructor nu este necesară deoarece limbajul C# are asociat un mecanism special care distruge automat orice obiect care nu mai este folosit („garbage collector”). Exemplu:

class Distrus

{ public string nume;

public Distrus(string n)

{

nume = n;

}

~Distrus()

{

Console.Write("\nObiectul " + nume + " a fost distruuus!");

}

}

class Program

{ static void Main(string[] args)

{ //setam setul de caractere la UTF8 => se pot afișa diacritice în consolă

Console.OutputEncoding = UTF8Encoding.UTF8;

//definim câteva obiecte de tip Distrus

Distrus d1 = new Distrus("d1");

Distrus d2 = new Distrus("d2");

Distrus d3 = new Distrus("d3");

Distrus d4 = new Distrus("d4");

Distrus d5 = new Distrus("d5");

Distrus d6 = new Distrus("d6");

//"uităm" câteva obiecte folosind referințele d1, d2, d3 pentru alte obiecte

d1 = new Distrus("d1v2");

d2 = new Distrus("d2v2");

d3 = new Distrus("d3v2");

//gata

Console.Write("(aici s-a terminat programul)\n");

}

}

Se va afișa (se observă că obiectele au fost distruse după terminarea programului):

**(aici s-a terminat programul)**

**Obiectul d3v2 a fost distruuus!**

**Obiectul d2v2 a fost distruuus!**

**Obiectul d1v2 a fost distruuus!**

**Obiectul d6 a fost distruuus!**

**Obiectul d5 a fost distruuus!**

**Obiectul d4 a fost distruuus!**

**Obiectul d3 a fost distruuus!**

**Obiectul d2 a fost distruuus!**

**Obiectul d1 a fost distruuus!Press any key to continue . . .**